สุทธิศักดิ์ พงศ์ธนาพาณิช

SOFT MAIL

Advanced BASIC Programming Technique

ตอน การเข้ารหัสไฟล์สำหรับ VB/Win ด้วย CIPHER.DLL

เสนอไฟล์ไลบรารี CIPHER.DLL ซึ่งประกอบด้วยฟังก์ชันด้านการเข้ารหัสและถอดรหัสข้อมูลของไฟล์

จากตอนที่ผ่านมาผู้เขียนได้เขียนไฟล์ไลบรารี BENMARK.DLL ซึ่งประกอบด้วยฟังก์ชันด้านเวลาสำหรับการทำ benchmark โค้ดของโปรแกรมสำหรับในบทความตอนนี้ผู้เขียนจะขอเสนอไฟล์ไลบรารี CIPHER.DLL ซึ่งประกอบด้วยฟังก์ชันด้านการเข้ารหัสและถอดรหัสข้อมูลของไฟล์ ซึ่งเป็นฟังก์ชันที่ทำงานในระดับบิต ดังนั้นการสร้างฟังก์ชันดังกล่าวให้อยู่ในรูปแบบไฟล์ไลบรารี ก็จะช่วยให้การทำงานรวดเร็วขึ้น เพราะภาษาที่ใช้ในการสร้างไฟล์ไลบรารีส่วนใหญ่จะสนับสนุนการทำงานระดับบิตและสนับสนุนการใช้งานพอยน์เตอร์ ซึ่งจะสามารถประมวลผลได้รวดเร็วกว่าคำสั่งด้านการจัดการสตริงของ VB/Win มากทีเดียว โดยจากการทดสอบบนเครื่องของผู้เขียน ซึ่งเป็นเครื่อง Pentium-90, RAM 16 MB และ WfW 3.11 ปรากฏว่าสามารถทำการเข้ารหัสหรือถอดรหัสได้ด้วยความเร็ว 50,000 ไบต์ต่อวินาทีเลยทีเดียว

สำหรับผู้อ่านที่ต้องการไฟล์ไลบรารี CIPHER.DLL หรือ BENMARK.DLL ที่เป็นโค้ด 32 บิต สนับสนุน VB/Win - 32 บิต ซึ่งทำงานภายใต้ระบบปฏิบัติการวินโดวส์ 95 หรือเอ็นทีก็ต้องรอกันอีกสัก 1 - 2 เดือนครับ เพราะผู้เขียนกำลังพัฒนา ถ้าหากผ่านการทดสอบเป็นที่เรียบร้อย ผู้เขียนก็จะแจกให้ฟรีผ่านทางนิตยสารไมโครฯนี้แน่นอนครับ

ก่อนที่ผู้อ่านจะใช้โปรแกรมตัวอย่าง DCIPHER.EXE ประจำบทความตอนนี้ ผู้อ่านต้องทำการคัดลอกไฟล์ไลบรารี CIPHER.DLL และ BENMARK.DLL ไปไว้ในไดเรคทอรีย่อย SYSTEM ของวินโดวส์เสียก่อน (ซึ่งโดยปกติจะมีชื่อเส้นทางเต็ม ๆ ดังนี้ C:\WINDOWS\SYSTEM) ส่วนวิธีการใช้งานฟังก์ชันของไฟล์ไลบรารี BENMARK.DLL ผู้อ่านก็สามารถติดตามได้ในฉบับที่ผ่านมาครับ และสามารถสั่งซื้อซอร์สโค้ดได้ทางซอฟต์เมล์ครับ

เบื้องต้นเกี่ยวกับการเข้ารหัสข้อมูล

ในยุคปัจจุบันการเข้ารหัสข้อมูลนับได้ว่ามีความสำคัญเป็นอย่างมาก เนื่องจากมีการส่งผ่านข้อมูลทางเครือข่ายกันมาก ไม่ว่าจะเป็น เครือข่ายเน็ตเวอร์ก เครือข่ายอินเทอร์เน็ต หรือเครือข่ายโทรศัพท์ เป็นต้น ดังนั้นความปลอดภัยของข้อมูลจึงมีความสำคัญเป็นอย่างสูง เนื่องจากการขโมยข้อมูลผ่านทางเครือข่ายนั้น สามารถทำได้โดยไม่ยากนัก ยกตัวอย่างเช่น เครือข่ายเน็ตเวอร์ก LAN เราสามารถตรวจสอบและขอดูข้อมูลที่ส่งผ่านระบบเครือข่ายได้โดยการใช้อุปกรณ์ที่เรียกว่า LAN ANALYZER ดังนั้นถ้าหากเราทำการเข้ารหัสข้อมูลก่อนการส่งก็จะทำให้ข้อมูลดังกล่าวจะต่างจากข้อมูลต้นฉบับอย่างสิ้นเชิง

ในปัจจุบันได้มีการพัฒนาเทคนิคการเข้ารหัสข้อมูลกันอย่างหลากหลาย ซึ่งบางเทคนิคก็ถูกพัฒนาเพื่อการเข้ารหัสข้อมูลโดยทั่ว ๆ ไป แต่บางเทคนิคก็ถูกพัฒนา

สำหรับการเข้ารหัสข้อมูลที่ส่งผ่านทางเครือข่ายโทรศัพท์เท่านั้น แต่เนื่องจากการอธิบายถึงเทคนิคต่าง ๆไม่อยู่ในขอบเขตของบทความตอนนี้ ดังนั้นผู้เขียนก็จะอธิบายเฉพาะแนวคิดพื้นฐานสำหรับการเข้ารหัสข้อมูลเพื่อเป็นแนวทางให้กับผู้อ่านเท่านั้น

โดยทั่วไปการเข้ารหัสข้อมูลจะหมายถึง การนำรหัสผ่าน (password) ที่กำหนด มากระทำกับข้อมูลแต่ละไบต์ด้วยวิธีทางคณิตศาสตร์ เพื่อให้ข้อมูลที่ได้แตกต่างไปจากเดิม ดังนั้นวิธีการเข้ารหัสข้อมูลที่ถูกต้องจะต้องสามารถถอดรหัสข้อมูลได้อย่างถูกต้อง ถ้าหากใช้วิธีการคณิตศาสตร์เหมือนเดิมกระทำกับข้อมูลแต่ละไบต์ โดยรหัสผ่านเดิม ซึ่งโดยทั่วไปทุก ๆ เทคนิคการเข้ารหัสข้อมูลมักจะใช้ตัวปฏิบัติการทางตรรกะ Exclusive OR (XOR) ในการเข้ารหัสหรือถอดรหัสข้อมูล ซึ่งตัวปฏิบัติการทางตรรกะ XOR มีคุณสมบัติ ดังนี้

```
TRUE TABLE FOR Exclusive OR
0 XOR 0 = 0
1 XOR 0 = 1
0 XOR 1 = 1
1 XOR 1 = 0
```

ดังนั้นจะเห็นว่าถ้าหากเราทำการ XOR บิตใด ๆ 2 ครั้ง ด้วยบิต 0 หรือ 1 ที่เหมือน ๆ กัน ก็จะได้กลับค่าของบิตเดิม (ก่อนการ XOR) เสมอ ซึ่งจากหลักการนี้ ผู้อ่านก็จะเห็นว่าถ้าหากเราทำการ XOR แต่ละไบต์ของข้อมูล จำนวน 2 ครั้ง ด้วยค่าแอสกีเดิม ผลลัพธ์ที่ได้ก็จะเป็นข้อมูลที่มีค่าเหมือนเดิมก่อนการ XOR ทุกประการ เช่น

```
1st XOR     &HA XOR &H9 = 1010 XOR 1001 = 0011
            หรือ    &H3 (3D)
2nd XOR     &H3 XOR &H9 = 0011 XOR 1001 = 1010
            หรือ    &HA (10D)
```

จากสมการคณิตศาสตร์ข้างต้น สมมติว่าข้อมูลมีค่าเท่ากับแอสกี 10 (0AH) และรหัสผ่านมีค่าเท่ากับ 9 (09H) เมื่อทำการเข้ารหัสด้วย XOR ครั้งแรกก็จะได้ผลลัพธ์เท่ากับ 3 (03H) จากนั้นเมื่อเราต้องการถอดรหัสเราก็ต้องนำข้อมูลที่ถูกเข้ารหัสเท่ากับ 3 มาทำการ XOR อีกครั้งด้วยรหัสผ่านเดิม (9) จะได้ผลลัพธ์เท่ากับ 10 หรือข้อมูลก่อนการเข้ารหัสทุกประการ

จากข้อมูลเบื้องต้น ผู้อ่านจะเห็นว่าเทคนิคการเข้ารหัสข้อมูลนั้นไม่ยาก และก็เช่นเดียวกันแนวคิดข้างต้นก็สามารถทำให้การแกะรหัสผ่านนั้นค่อนข้างง่ายไปด้วยเช่นเดียวกัน ดังนั้นในปัจจุบันจึงได้มีการเพิ่มลูกเล่นต่าง ๆ มากมายให้กับการเข้ารหัส เช่น การสร้างชุดตัวเลขสุ่มสำหรับการเข้ารหัส การนำชุดรหัสผ่านมาสร้างรหัสอีกครั้งก่อนการเข้ารหัส เป็นต้น ซึ่งด้วยเทคนิคใหม่ ๆ เหล่านี้ก็จะเพิ่มความซับซ้อนในการเข้ารหัสมากขึ้น จึงทำให้การแกะรหัสผ่านทำได้ยากมากขึ้น

ไฟล์ไลบรารี CIPHER.DLL

ฟังก์ชันการเข้ารหัสและถอดรหัสของไฟล์ไลบรารี CIPHER.DLL ก็ใช้เทคนิคเช่นเดียวกับที่กล่าวมาแล้วข้างต้น โดยการใช้ตัวปฏิบัติการทางตรรกะ XOR ในการกระทำกับข้อมูลเพื่อเข้ารหัสและถอดรหัส แต่ผู้เขียนก็ได้เพิ่มอัลกอริธึมในการสุ่มชุดรหัสผ่านขึ้นมาใหม่ โดยอาศัยข้อมูลจากชุดรหัสผ่านที่ผู้อ่านกำหนด ขนาดของไฟล์ ขนาดของบล็อกข้อมูล และอื่น ๆ จากนั้นจึงนำชุดรหัสผ่านใหม่ที่ได้มาทำการเข้ารหัสหรือถอดรหัส ดังนั้นจึงทำให้ข้อมูลที่ได้จึงมีความปลอดภัยมากกว่า เพราะถึงแม้ว่าโปรแกรมเมอร์บางท่านอาจจะสามารถถอดรหัสผ่านที่ใช้ในการเข้ารหัสได้ก็ตาม แต่จะไม่สามารถนำมาค้นหารหัสผ่านที่แท้จริงได้เลย เนื่องจากข้อมูลที่ใช้ในการสุ่มรหัสผ่านใหม่ จะไม่ถูกจัดเก็บลงในไฟล์แต่จะได้มาจากสมการทางคณิตศาสตร์ที่แน่นอนนั่นเอง

สำหรับฟังก์ชันต่าง ๆ ที่มีในไฟล์ไลบรารี CIPHER.DLL ก็มีดังต่อไปนี้

```
Public Declare Function cphCloseFile Lib_
            "CIPHER.DLL"(ByVal Force CloseFlag_
            As Integer) As Integer
Public Declare Function cphCipherDataBlock_
            Lib "CIPHER.DLL"(ByVal BlockNum As_
            Long) As Integer
Public Declare Function cphDecipherDataBlock_
            Lib "CIPHER.DLL"(ByVal BlockNum As_
            Long) As Integer
Public Declare Function cphErrorMsg Lib_
            "CIPHER.DLL"(ByVal ErrCode As_
            Integer) As String
Public Declare Function cphGetBlocks Lib_
            "CIPHER.DLL" () As Long
Public Declare Function cphGetBlockSize Lib_
            "CIPHER.DLL" () As Integer
Public Declare Function cphGetCopyright Lib_
            "CIPHER.DLL" () As String
Public Declare Function cphGetNextByte Lib_
            "CIPHER.DLL" (ByVal BlockNum As_
            Long) As Long
Public Declare Function cphGetPassword Lib_
            "CIPHER.DLL" () As String
Public Declare Function cphGetPercentProgress_
            Lib "CIPHER.DLL" (ByVal BlockNum_
            As Long) As Single
Public Declare Function cphGetVersion Lib_
            "CIPHER.DLL" () As Long
Public Declare Function cphOpenFile Lib_
            "CIPHER.DLL" (filename$) As Integer
Public Declare Function cphResetBlockSize Lib_
            "CIPHER.DLL" () As Integer
Public Declare Function cphSetBlockSize Lib_
            "CIPHER.DLL" (ByVal blksize As_
            Integer) As Integer
Public Declare Function cphSetPassword Lib_
            "CIPHER.DLL" (ByVal password As_
            String) As Integer
```

ส่วนรายละเอียดเกี่ยวกับวิธีใช้งานแต่ละฟังก์ชันผู้เขียนก็จะอธิบายในหัวข้อถัดไป แต่เพื่อให้ผู้อ่านได้เห็นถึงภาพกว้าง ๆ ในการเข้ารหัสและถอดรหัสข้อมูลด้วยไฟล์ไลบรารี CIPHER.DLL ผู้เขียนก็จะขอแสดงตัวอย่างของโค้ดและขั้นตอนในการเขียนโค้ดดังต่อไปนี้

การเข้ารหัสและถอดรหัสข้อมูล (cipher and decipher coding)

1. กำหนดรหัสผ่าน (password) เพื่อใช้ในการเข้ารหัส ซึ่งเป็นข้อความสตริงมีความยาวไม่เกิน 16 ตัวอักษร เช่น ABDos1-12 เป็นต้น ดังนี้

```
Dim pwd AS STRING

pwd = "ABDos1-12"
ErrCode% = cphSetPassword(pwd)
' if error read it's description.
If ErrCode% Then
  MsgBox cphErrorMsg(ErrCode%),16,Me.Caption
End If
```

สำหรับการถอดรหัสข้อมูล ผู้อ่านต้องกำหนดรหัสผ่านให้เหมือนกับการเข้ารหัสข้อมูลของแต่ละไฟล์ทุกประการ ซึ่งตัวอักษรทุกตัวจะถูกตีความเป็นชนิด case sensitive ดังนั้นตัวอักษรพิมพ์เล็ก a จึงไม่เหมือนกับตัวอักษรพิมพ์ใหญ่ A เป็นต้น

2. กำหนดขนาดของบล็อกข้อมูล (data block) ในการอ่านหรือเขียนลงในไฟล์ ซึ่งจะมีค่าเป็นตัวเลขจำนวนเต็มตั้งแต่ 1 ถึง 24 บล็อก โดยที่ขนาดของบล็อกจะเท่ากับ 1,024 ไบต์ (1 กิโลไบต์) ดังนี้

```
Dim blksize As Integer

ErrCode% = cphSetBlockSize(blksize)
If ErrCode% Then
    MsgBox cphErrorMsg(ErrCode%),16, Me.Caption
End If
```

ถ้าหากผู้อ่านไม่กำหนดขนาดของบล็อกข้อมูล เมื่อทำการเข้ารหัสหรือถอดรหัส ไฟล์ไลบรารี CIPHER.DLL ก็จะกำหนดให้มีขนาดเท่ากับ 4 กิโลไบต์โดยอัตโนมัติ

ข้อควรระวัง

สำหรับการกำหนดขนาดของบล็อกข้อมูลให้กับการถอดรหัสข้อมูลนั้น มีความสำคัญเป็นอย่างมาก ดังนั้นผู้อ่านต้องกำหนดให้มีขนาดเท่ากับขนาดบล็อกข้อมูลในขั้นตอนการเข้ารหัสข้อมูล เพราะตัวเลขขนาดของบล็อกข้อมูลจะถูกนำไปใช้ในการสุ่มรหัสผ่านนั่นเอง

3. เลือกไฟล์ที่ต้องการเข้ารหัสและเปิดไฟล์ดังกล่าวเอาไว้

```
Dim srcfilename$

' open selected file.
ErrCode% = cphOpenFile(srcfilename$)
If ErrCode% Then
    MsgBox cphErrorMsg(ErrCode%), 16, Me.Caption
End If
```

4 เริ่มต้นทำการเข้ารหัสข้อมูลหรือถอดรหัสข้อมูลของไฟล์

```
  Dim p$, ErrCode%, i As Long, DataBlocks_
  As Long

' get numberof data blocks.
DataBlocks = cphGetBlocks()

' start cipher opeation.
For i = 1 To DataBlocks
  ErrCode% = cphCipherDataBlock(i)
  ErrCode% = cphDecipherDataBlock(i)
  If ErrCode% Then
    MsgBox cphErrorMsg(ErrCode%), 16, Me.Caption
     Exit For
    End If
    DoEvents
    If IntFlag% Then Exit For
  Next i
```

จากโค้ดข้างต้น จากขั้นตอนที่ 3 เมื่อผู้อ่านเปิดไฟล์ที่เลือกด้วยฟังก์ชัน cphOpenFile ไฟล์ไลบรารี CIPHER.DLL ก็จะทำการคำนวณหาจำนวนบล็อกของข้อมูลที่จะต้องถูกอ่านและบันทึกในระหว่างการเข้ารหัสและถอดรหัสข้อมูลโดยอัตโนมัติ ซึ่งผู้อ่านต้องอ่านค่าตัวเลขดังกล่าวด้วยฟังก์ชันดังนี้

```
DataBlocks = cphGetBlocks()
```

จากนั้นผู้อ่านก็ต้องทำการวนลูปตามจำนวนบล็อกที่ได้ เพื่อทำการเข้ารหัสหรือถอดรหัสข้อมูลทุก ๆ ไบต์ โดยถ้าหากผู้อ่านเขียนโค้ดเพื่อทำการเข้ารหัสข้อมูลก็ให้เลือกโค้ดบรรทัด

```
ErrCode% = cphCipherDataBlock(i)
```

แต่ถ้าหากผู้อ่านเขียนโค้ดเพื่อทำการถอดรหัสข้อมูลก็ให้เลือกโค้ดบรรทัด

```
ErrCode% = cphDecipherDataBlock(i)
```

5. ปิดไฟล์ที่ถูกเข้ารหัสข้อมูล

```
' close file
ErrCode% = cphCloseFile(False)
If ErrCode% Then
  MsgBox cphErrorMsg(ErrCode%), 16, Me.Caption
  ' if error force it to closed.
  ErrCode% = cphCloseFile(True)
End If
```

ก่อนที่ผู้อ่านจะสามารถเปิดไฟล์อื่น ๆ เพื่อทำการเข้ารหัสหรือถอดรหัสข้อมูล ผู้อ่านต้องทำการปิดไฟล์ก่อนหน้าทุกครั้ง เพราะไฟล์ไลบรารี CIPHER.DLL สามารถอ่านบัฟเฟอร์ได้สำหรับ 1 ไฟล์เท่านั้น

รายละเอียดฟังก์ชันของไฟล์ไลบรารี CIPHER.DLL

การอธิบายถึงรายละเอียดและวิธีการใช้งานฟังก์ชันทั้งหมดของไฟล์ไลบรารี CIPHER.DLL โดยจะเรียงลำดับแต่ละฟังก์ชันตามกลุ่มของฟังก์ชันและตามลำดับตัวอักษร ดังต่อไปนี้

จากหัวข้อข้างต้นผู้อ่านก็ทราบถึงวิธีการเข้าโค้ดเพื่อทำการเข้ารหัสหรือถอดรหัสข้อมูลกันแล้ว ดังนั้นในหัวข้อนี้ผู้อ่านก็จะทำ

cphGetVersion

Declaration
```
Public Declare Function cphGet Version Lib "CIPHER.DLL" () As Long
```
Description รายงานหมายเลขเวอร์ชันของไฟล์ไลบรารี CIPHER.DLL

Parameters ไม่มี

Example
```
lVer& = cphGetVersion()
r$ = Format$(Hex$(lVer&), "00000000")
' major.minor.revision
verMajor% = Val("&H" & Mid$(r$, 3, 2))
verMinor% = Val("&H" & Mid$(r$, 5, 2))
verRevision% = Val("&H" & Mid$(r$, 7, 2))

p$ = "DLL version : " & verMajor% & "." & verMinor% & "." & _
                        verRevision% & vbCrLf
p$ = p$ & "Author : " & cphGetCopyright() & vbCrLf
```

Comment ค่าที่ส่งกลับโดยฟังก์ชันนี้จะบอกถึงหมายเลขเวอร์ชันของไฟล์ไลบรารี โดยผู้อ่านสามารถดูรายละเอียดได้จากตอนที่ผ่านมา เรื่อง การทำ BENCHMARK สำหรับ VB/Win ด้วย BENMARK.DLL

cphGetCopyright

Declaration
```
Public Declare Function btrGetCopyright  Lib "BENMARK.DLL" () As String
```
Description รายงานชื่อผู้เขียนและการสงวนลิขสิทธิ์ซอฟต์แวร์

Parameters ไม่มี

Example
```
Debug.Print "Author : "; btrGetCopyright()
```

Comment เนื่องจากฟังก์ชันทั้งหมดของไลบรารีได้ถูกเขียนขึ้นมาสำหรับ VB/Win โดยเฉพาะ ดังนั้นในกรณีที่ฟังก์ชันต้องการตัวแปรชนิดสตริงหรือมีการส่งค่ากลับในรูปแบบสตริง ก็ให้ผู้อ่านใช้ตัวแปรสตริงทั่วไปของ VB/Win เป็นตัวแปรให้กับฟังก์ชันหรือรับค่าจากฟังก์ชันก็ได้ เพราะภายในไลบรารีได้ทำการแปลงรูปแบบของสตริงให้คอมแพตทิเบิลกับสตริงของ VB/Win โดยอัตโนมัติอยู่แล้ว

cphCloseFile

Declaration
```
Public Declare Function cphCloseFile Lib "CIPHER.DLL" _
(ByVal ForceCloseFlag As Integer) As Integer
```
Description กำหนดให้มีการปิดไฟล์ที่ถูกเปิดก่อนหน้านี้ และ ยกเลิกบัฟเฟอร์สำหรับข้อมูล

Parameters `ForceCloseFlag` สามารถมีค่าได้ดังนี้

True หมายถึง ปิดไฟล์โดยไม่สนใจว่าจะเกิด ข้อผิดพลาดขึ้นหรือไม่

False หมายถึง ปิดไฟล์โดยสนใจข้อผิดพลาดที่เกิดขึ้น

Example
```
' close file
ErrCode% = cphCloseFile(False)
If ErrCode% Then
      MsgBox cphErrorMsg(ErrCode%), 16,Me.Caption
      ' if error force it to closed.
      ErrCode% = cphCloseFile(True)
End If
```

Comment ในกรณีที่ ForceCloseFlag มีค่าเท่ากับ True ค่าที่ได้จากฟังก์ชันนี้จะเท่ากับ 0 เสมอ ทั้งนี้เนื่องจากฟังก์ชันจะบังคับให้มีการปิดไฟล์ โดยไม่คำนึงถึงข้อผิดพลาดที่เกิดขึ้นในระหว่างการทำงาน แต่ถ้าหาก ForceCloseFlag มีค่าเท่ากับ False ถ้าหากมีข้อผิดพลาดเกิดขึ้นในขณะกำลังปิดไฟล์ ฟังก์ชันก็จะรายงานค่าความผิดพลาดกลับทันที และถ้าหากการปิดไฟล์สำเร็จฟังก์ชันก็จะรายงานค่ากลับเป็น 0

cphCipherDataBlock

Declaration
```
Public Declare Function cphCipherDataBlock Lib "CIPHER.DLL" _
(ByVal BlockNum As Long) As Integer
```
Description ทำการเข้ารหัสข้อมูลทุกไบต์ของบล็อกที่กำหนด

Parameters `BlockNum` เลขจำนวนเต็ม long กำหนดหมายเลขบล็อกข้อมูล ซึ่งสามารถมีค่าได้ตั้งแต่ 1 ถึง 2,147,483,647 บล็อก

Example
```
Dim ErrCode%
' cipher data block number 1.
ErrCode% = cphCipherDataBlock(1)
If ErrCode% Then
      MsgBox cphErrorMsg(ErrCode%), 16, Me.Caption
      Exit For
End If
```

Comment สำหรับจำนวนบล็อกของข้อมูลที่จะทำการเข้ารหัสนั้น ผู้อ่านสามารถอ่านค่าได้จากฟังก์ชัน cphGetBlocks ซึ่งไฟล์ไลบรารี CIPHER.DLL จะทำการคำนวณให้โดยอัตโนมัติ สำหรับค่าที่คืนกลับโดยฟังก์ชันนี้ มีดังต่อไปนี้

0 หมายถึง ฟังก์ชันทำงานได้สำเร็จสมบูรณ์

> 1 หมายถึง ฟังก์ชันทำงานล้มเหลว ซึ่งหมายเลขข้อผิดพลาดที่สามารถเป็นไปได้ มีดังต่อไปนี้

32759 CIPHER.DLL ไม่สามารถอ้างหน่วยความจำภายในสำหรับสตริงได้

32766 ถ้าหากค้นหาไฟล์ที่กำหนดไม่พบ หรือไฟล์มีความยาวเท่ากับ 0 ไบต์หรืออีกนัยหนึ่งหมายถึง ไฟล์ที่กำหนดยังไม่ถูกเปิด

cphDecipherDataBlock

Declaration
```
Public Declare Function cphDecipherDataBlock Lib "CIPHER.DLL" _
(ByVal BlockNum As Long) As Integer
```
Description ทำการถอดรหัสข้อมูลทุกไบต์ของบล็อกที่กำหนด

Parameters `BlockNum` เลขจำนวนเต็ม long กำหนดหมายเลขบล็อกข้อมูล ซึ่งสามารถมีค่าได้ตั้งแต่ 1 ถึง 2,147,483,647 บล็อก

Example
```
Dim ErrCode%
' decipher data block number 15.
ErrCode% = cphDecipherDataBlock(15)
If ErrCode% Then
      MsgBox cphErrorMsg(ErrCode%), 16, Me.Caption
      Exit For
End If
```

Comment สำหรับจำนวนบล็อกของข้อมูลที่จะทำการถอดรหัสนั้น ผู้อ่านสามารถอ่านค่าได้จากฟังก์ชัน cphGetBlocks ซึ่งไฟล์ไลบรารี CIPHER.DLL จะทำการคำนวณให้โดยอัตโนมัติ สำหรับค่าที่คืนกลับโดยฟังก์ชันนี้ มีดังต่อไปนี้

0 หมายถึง ฟังก์ชันทำงานได้สำเร็จสมบูรณ์

> 1 หมายถึง ฟังก์ชันทำงานล้มเหลว ซึ่งหมายเลขข้อผิดพลาดที่สามารถเป็นไปได้ มีดังต่อไปนี้

32759 CIPHER.DLL ไม่สามารถอ้างหน่วยความจำภายในสำหรับสตริงได้

32766 ถ้าหากค้นหาไฟล์ที่กำหนดไม่พบ หรือไฟล์มีความยาวเท่ากับ 0 ไบต์หรืออีกนัยหนึ่งหมายถึง ไฟล์ที่กำหนดยังไม่ถูกเปิด

cphErrorMsg

Declaration
```
Public Declare Function cphErrorMsg Lib "CIPHER.DLL" _
(ByVal ErrCode As Integer) As String
```
Description รายงานข้อความรายละเอียดของข้อผิดพลาดหมายเลขต่าง ๆ

Parameters `ErrCode` หมายเลข ID ของข้อผิดพลาดซึ่งสามารถมีค่าได้ตั้งแต่ 1 ถึง 32,767

Example
```
MsgBox cphErrorMsg(ErrCode%), 16, Me.Caption
```
Comment สำหรับค่าที่คืนกลับโดยฟังก์ชันนี้จะเป็นข้อความสตริง เช่น File Not Found. เป็นต้น

cphGetBlocks

Declaration
```
Public Declare Function cphGetBlocks Lib "CIPHER.DLL" () As Long
```
Description รายงานจำนวนบล็อกข้อมูลทั้งหมดของไฟล์

Parameters ไม่มี

Example
```
Dim DataBlocks As Long
' get numberof data blocks.
DataBlocks = cphGetBlocks()
```
Comment จำนวนบล็อกข้อมูลของไฟล์นั้น จะคำนวณโดยใช้สมการดังนี้

```
DataBlocks = (Length_Of_File - 1) \ (Block_Size * 1024) + 1&
```

cphGetBlockSize

Declaration
```
Public Declare Function cphGetBlockSize Lib "CIPHER.DLL" () As Integer
```
Description อ่านขนาดของบล็อกข้อมูล

Parameters ไม่มี

Example
```
Dim BlockSize%
BlockSize% = cphGetBlockSize()
```
Comment ขนาดของบล็อกข้อมูลสามารถมีค่าได้ตั้งแต่ 1 ถึง 24 บล็อก โดยปกติถ้าหากผู้อ่านยังไม่มีการกำหนดขนาดบล็อกข้อมูล CIPHER.DLL ก็จะกำหนดให้มีค่าเท่ากับ 4 โดยอัตโนมัติ

cphGetNextByte

Declaration
```
Public Declare Function cphGetNextByte Lib "CIPHER.DLL" _
(ByVal BlockNum As Long) As Long
```
Description รายงานไบต์เริ่มต้นของบล็อกข้อมูลของไฟล์ที่จะถูกทำการเข้ารหัสหรือถอดรหัสข้อมูล

Parameters `BlockNum` เลขจำนวนเต็ม long กำหนดหมายเลขบล็อกข้อมูล ซึ่งสามารถมีค่าได้ตั้งแต่ 1 ถึง 2,147,483,647 บล็อก

Example
```
' report first byte of block 15th.
BytePos& = cphGetNextByte(15)
Print  "Working on byte "; BytePos&
```
Comment โดยปกติเราจะอ่านไบต์เริ่มต้นของบล็อกข้อมูลของไฟล์ที่จะถูกทำการเข้ารหัสหรือถอดรหัสข้อมูล เพื่อรายงานสถานะความคืบหน้าของการทำงาน ดังนั้นผู้อ่านจึงควรใช้ฟังก์ชันนี้ในวงลูปที่ทำการเข้ารหัสหรือถอดรหัสข้อมูล

cphGetPassword

Declaration `Public Declare Function cphGetPassword Lib "CIPHER.DLL" () As String`

Description อ่านรหัสผ่านที่ถูกกำหนดให้กับ CIPHER.DLL

Parameters ไม่มี

Example
```
' read valid password.
Dim pwd As String
pwd = cphGetPassword()
```

Comment สำหรับค่าที่คืนกลับโดยฟังก์ชันนี้ จะเป็นสตริงของรหัสผ่านมีความยาวตั้งแต่ 0 ถึง 16 ตัวอักษร

cphGetPercentProgress

Declaration
```
Public Declare Function cphGetPercentProgress  Lib "CIPHER.DLL" _
(ByVal BlockNum As Long) As Single
```

Description รายงานเปอร์เซ็นต์ความคืบหน้าของการเข้ารหัสหรือถอดรหัสข้อมูล

Parameters `BlockNum` เลขจำนวนเต็ม long กำหนดหมายเลขบล็อกข้อมูล
ซึ่งสามารถมีค่าได้ตั้งแต่ 1 ถึง 2,147,483,647 บล็อก

Example
```
' report percent progress of block 15th.
Dim PercentProgress As Integer

PercentProgress = cphGetPercentProgress(15)
Print "Working Progress "; BytePos&; " %"
```

Comment สำหรับค่าที่คืนกลับโดยฟังก์ชันนี้ มีดังต่อไปนี้ มีค่าตั้งแต่ 0 ถึง 100% ซึ่งผู้อ่านสามารถใช้ค่าที่ได้จากฟังก์ชันนี้ในการรายงานผลความคืบหน้าได้อีกรูปแบบ (เปอร์เซ็นต์) และผู้อ่านยังสามารถนำไปประยุกต์ใช้กับ PictureBox เพื่อรายงานผลในรูปแบบกราฟิกได้อีกด้วย

cphOpenFile

Declaration `Public Declare Function cphOpenFile Lib "CIPHER.DLL" (filename$) As Integer`

Description ทำการเปิดไฟล์ที่ต้องการเข้ารหัสหรือถอดรหัสข้อมูล และทำการอ้างบัฟเฟอร์สำหรับข้อมูลตามขนาดที่กำหนดด้วย

Parameters `filename$` ชื่อของไฟล์ที่กำหนด (สามารถรวมชื่อเส้นทางได้ด้วย)

Example
```
' open selected file.
ErrCode% =  cphOpenFile(srcfilename$)
OpenFlag% = (ErrCode% = False)
If ErrCode% Then
      MsgBox cphErrorMsg(ErrCode%), 16, Me.Caption
End If
```

Comment สำหรับค่าที่คืนกลับโดยฟังก์ชันนี้ มีดังต่อไปนี้

0 หมายถึง ฟังก์ชันทำงานได้สำเร็จสมบูรณ์

> 1 หมายถึง ฟังก์ชันทำงานล้มเหลว ซึ่งหมายเลขข้อผิดพลาดที่สามารถเป็นไปได้ มีดังต่อไปนี้

32760 ถ้าหากรหัสผ่านยังไม่ถูกกำหนด

32762 ถ้าหาก CIPHER.DLL ไม่สามารถอ้างหมายเลข handle ของไฟล์

32766 ถ้าหากค้นหาไฟล์ที่กำหนดไม่พบ หรือไฟล์มีความยาวเท่ากับ 0 ไบต์

32767 ถ้าหากไฟล์ก่อนหน้ายังคงถูกเปิด

cphResetBlockSize

Declaration `Public Declare Function cphResetBlockSize Lib "CIPHER.DLL" () As Integer`

Description ทำการกำหนดขนาดของบล็อกข้อมูลให้เท่ากับ 4 (4 KB)

Parameters ไม่มี

Example
```
ErrCode% = cphResetBlockSize()
If ErrCode% Then
     MsgBox cphErrorMsg(ErrCode%), 16, Me.Caption
End If
```

Comment สำหรับค่าที่คืนกลับโดยฟังก์ชันนี้ มีดังต่อไปนี้

0 หมายถึง ฟังก์ชันทำงานได้สำเร็จสมบูรณ์

> 1 หมายถึง ฟังก์ชันทำงานล้มเหลว ซึ่งหมายเลขข้อผิดพลาดที่สามารถเป็นไปได้ มีดังต่อไปนี้

32767 ถ้าหากไฟล์ก่อนหน้ายังคงถูกเปิด

ฟังก์ชันนี้จะไม่สามารถใช้ได้ในขณะที่ไฟล์ยังคงถูกเปิดอยู่

cphSetBlockSize

Declaration
```
Public Declare Function cphSetBlockSize Lib "CIPHER.DLL" _
(ByVal blksize As Integer) As Integer
```

Description ทำการกำหนดขนาดของบล็อกข้อมูลตามที่กำหนด

Parameters `blksize` กำหนดขนาดของบล็อกข้อมูล ซึ่งสามารถมีค่าได้ตั้งแต่ 1 ถึง 24 KB

Example
```
' set data buffer to 20 KB.
ErrCode% = cphSetBlockSize(20)
If ErrCode% Then
     MsgBox cphErrorMsg(ErrCode%), 16, Me.Caption
End If
```

Comment สำหรับค่าที่คืนกลับโดยฟังก์ชันนี้ มีดังต่อไปนี้

0 หมายถึง ฟังก์ชันทำงานได้สำเร็จสมบูรณ์

> 1 หมายถึง ฟังก์ชันทำงานล้มเหลว ซึ่งหมายเลขข้อผิดพลาดที่สามารถเป็นไปได้ มีดังต่อไปนี้

32765 ถ้าหากขนาดของบล็อกข้อมูลมีค่าไม่อยู่ในช่วงตั้งแต่ 1 ถึง 24

32767 ถ้าหากไฟล์ก่อนหน้ายังคงถูกเปิด

ฟังก์ชันนี้จะไม่สามารถใช้ได้ในขณะที่ไฟล์ยังคงถูกเปิดอยู่

cphSetPassword

Declaration
```
Public Declare Function cphSetPassword Lib "CIPHER.DLL" _
(ByVal password As String) As Integer
```

Description ทำการกำหนดขนาดของบล็อกข้อมูลตามที่กำหนด

Parameters `blksize` กำหนดขนาดของบล็อกข้อมูล ซึ่งสามารถมีค่าได้ตั้งแต่ 1 ถึง 24 KB

Example
```
' set password to "Sutthisak".
Dim pwd As String

pwd = "Sutthisak"
ErrCode% = cphSetPassword(pwd)
If ErrCode% Then
MsgBox cphErrorMsg(ErrCode%), 16, Me.Caption
End If
```

Comment สำหรับค่าที่คืนกลับโดยฟังก์ชันนี้ มีดังต่อไปนี้

0 หมายถึง ฟังก์ชันทำงานได้สำเร็จสมบูรณ์

> 1 หมายถึง ฟังก์ชันทำงานล้มเหลว ซึ่งหมายเลขข้อผิดพลาดที่สามารถเป็นไปได้ มีดังต่อไปนี้

32764 ถ้าหากรหัสผ่านเป็นสตริงว่าง ("")

32767 ถ้าหากไฟล์ก่อนหน้ายังคงถูกเปิด

ฟังก์ชันนี้จะไม่สามารถใช้ได้ในขณะที่ไฟล์ยังคงถูกเปิดอยู่

โปรแกรม DCIPHER.EXE

สำหรับหัวข้อสุดท้ายที่ผู้เขียนจะกล่าวถึงในบทความตอนนี้ ก็จะเป็นวิธีการใช้งานโปรแกรม DCIPHER.EXE ส่วนรายละเอียดเกี่ยวกับวิธีการทำงาน วิธีการออกแบบโปรแกรม และซอร์สโค้ดทั้งหมดของโปรแกรม ผู้เขียนก็จะกล่าวถึงในฉบับต่อไปครับ

โปรแกรม DCIPHER เป็นโปรแกรมที่ประกอบด้วยฟอร์มเพียงฟอร์มเดียว (CIPHER.FRM) เมื่อผู้อ่านรันโปรแกรมภายใต้เอนไวรอนเมนท์ของ VB/Win ก็จะปรากฏฟอร์มดังในรูปที่ 1 โดยที่โปรแกรมนี้จะประกอบด้วย 2 เมนู ดังมีรายละเอียดดังต่อไปนี้

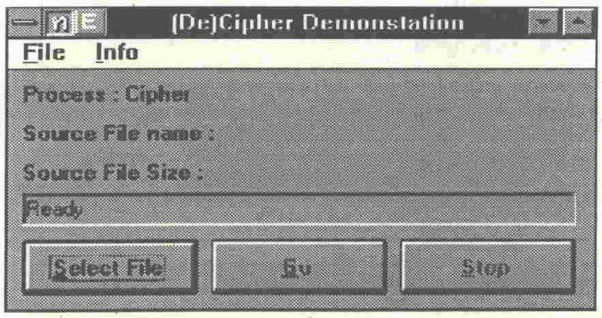
เมนู File ประกอบด้วยคำสั่งด้านการปิดไฟล์และเลือกการเข้ารหัสหรือถอดรหัส ดังนี้

คำสั่ง Manual Close File สำหรับการปิดไฟล์ ซึ่งโดยปกติจะถูกใช้เมื่อเกิดข้อผิดพลาดในขณะเปิดไฟล์ หรือในขณะที่กำลังทำการเข้ารหัสหรือถอดรหัสข้อมูล เป็นต้น

คำสั่ง Cipher กำหนดให้ทำการเข้ารหัสข้อมูล

คำสั่ง Decipher กำหนดให้ทำการถอดรหัสข้อมูล

คำสั่ง Exit ออกจากโปรแกรม



รูปที่ 1 แสดงโปรแกรม DCIPHER ในขณะเริ่มต้นการทำงาน

เมนู Info ประกอบด้วยคำสั่งด้านการข้อมูลค่าต่าง ๆ ที่กำหนดให้กับ CIPHER.DLL ดังนี้

คำสั่ง Get DLL Information สำหรับการอ่านเวอร์ชันของไฟล์ CIPHER.DLL

คำสั่ง Get Password สำหรับการอ่านรหัสผ่าน

คำสั่ง Set Password สำหรับการกำหนดรหัสผ่าน

คำสั่ง Reset Block Size สำหรับกำหนดให้บล็อกข้อมูลเท่ากับ 4 KB

คำสั่ง Get Block Size สำหรับอ่านขนาดของบล็อกข้อมูล

คำสั่ง Set Block Size สำหรับกำหนดให้บล็อกข้อมูลมีขนาดเท่าที่ต้องการ

ตามที่กล่าวมาแล้วในหัวข้อข้างต้น ถึงวิธีการเขียนโค้ดเพื่อทำการเข้ารหัสหรือถอดรหัสข้อมูล ดังนั้นผู้อ่านก็จะแนะนำวิธีการใช้งานโปรแกรมตามขั้นตอนข้างต้น ดังต่อไปนี้

1. กำหนดการปฏิบัติการ โดยเลือกคำสั่ง Cipher ถ้าหากต้องการเข้ารหัสข้อมูล หรือเลือกคำสั่ง Decipher ถ้าหากต้องการถอดรหัสข้อมูล โดยคำสั่งที่ถูกเลือกก็จะปรากฏเครื่องหมาย ✓ นำหน้าคำสั่งที่ถูกเลือกและชื่อของการปฏิบัติการก็จะปรากฏในช่อง Process ซึ่งเป็นแถวบนสุดของฟอร์ม

2. กำหนดรหัสผ่านสำหรับการเข้ารหัสหรือถอดรหัสข้อมูล ด้วยคำสั่ง Set Password ซึ่งจะปรากฏ Input box ดังในรูปที่ 2 และผู้อ่านสามารถกรอกรหัสผ่านได้ทั้งหมดไม่เกิน 16 ตัวอักษร

3. กำหนดขนาดของบล็อกข้อมูล ซึ่ง CIPHER.DLL จะใช้กำหนดขนาดของบัฟเฟอร์สำหรับการขนถ่ายข้อมูลจากไฟล์และหน่วยความจำ และที่สำคัญตัวเลขที่กำหนดขนาดของบล็อกข้อมูลยังจะถูกนำไปใช้ในการสุ่มรหัสผ่านภายในอีกครั้ง เพื่อนำรหัสผ่านภายในที่ได้มาทำการเข้ารหัสหรือถอดรหัสข้อมูล ดังนั้นในการถอดรหัสข้อมูลผู้อ่านต้องกำหนดให้ขนาดของบล็อกข้อมูลมีค่าเท่ากับขนาดที่ใช้เข้ารหัสข้อมูลทุกประการ เพราะมิฉะนั้นอาจจะทำให้การถอดรหัสข้อมูลไม่สำเร็จก็ได้ โดยในรูปที่ 3 เป็นการกำหนดให้ขนาดของบล็อกข้อมูลเท่ากับ 16 กิโลไบต์

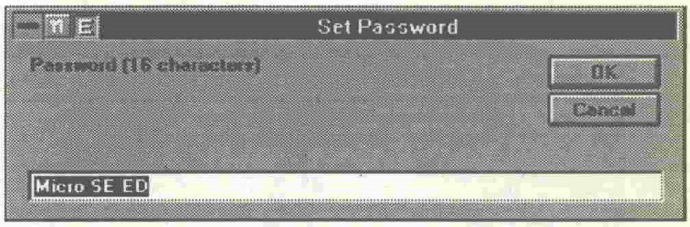
4. เลือกไฟล์ที่ต้องการเข้ารหัสหรือถอดรหัสข้อมูล โดยการคลิ๊กที่ปุ่มคำสั่ง Select File จากนั้นก็จะปรากฏไดอะล็อกบ็อกซ์ Select Source File ดังในรูปที่ 4 โดยเมื่อผู้อ่านเลือกไฟล์ที่ต้องการแล้วก็คลิ๊กที่ปุ่มคำสั่ง OK ด้านบนซ้ายของไดอะล็อกบ็อกซ์ ไฟล์ที่ถูกเลือกก็จะถูกเปิดทันที

เมื่อผู้อ่านเลือกไฟล์แล้ว โปรแกรมก็จะทำการเปิดไฟล์และอ่านขนาดของไฟล์ที่ถูกเลือก และนำมาแสดงผลชื่อไฟล์และขนาดของไฟล์ในช่อง Source File Name และ Source File Size ตามลำดับ ดังตัวอย่างในรูปที่ 5 เป็นการแสดงรายละเอียดของไฟล์ COMMAND.COM

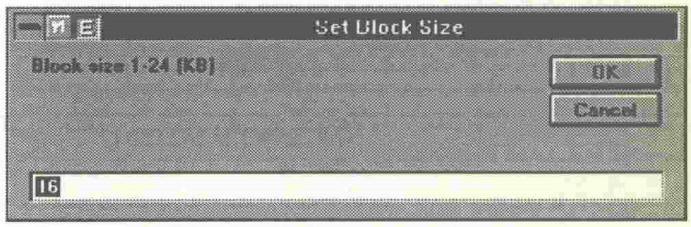
5. สุดท้ายก็เป็นขั้นตอนการเข้ารหัสหรือถอดรหัสข้อมูล โดยการคลิ๊กที่ปุ่มคำสั่ง Go เท่านั้น โปรแกรมก็จะปฏิบัติการตามที่กำหนดโดยอัตโนมัติ และในระหว่างการปฏิบัติการก็จะรายงานความคืบหน้าในช่องรายงานผล ซึ่งเป็นแถบแบบสามมิติด้านบนของแถบปุ่มคำสั่ง ดังตัวอย่างในรูปที่ 5 ข้างต้น

ในระหว่างการปฏิบัติการผู้อ่านสามารถยกเลิกการทำงานได้โดยการคลิ๊กที่ปุ่มคำสั่ง Stop เท่านั้น แต่ข้อควรระวังในการยกเลิกการทำงานก็คือ ข้อมูลของไฟล์ส่วนที่ถูกเข้ารหัสไปแล้วนั้นจะไม่สามารถกู้กลับมาได้ไม่ว่าด้วยวิธีใด ๆ ก็ตาม เพราะถึงแม้ว่าผู้อ่านจะทำการถอดรหัส ข้อมูลในส่วนที่ไม่ถูกเข้ารหัสก็จะมีค่าที่แตกต่างไปจากเดิม

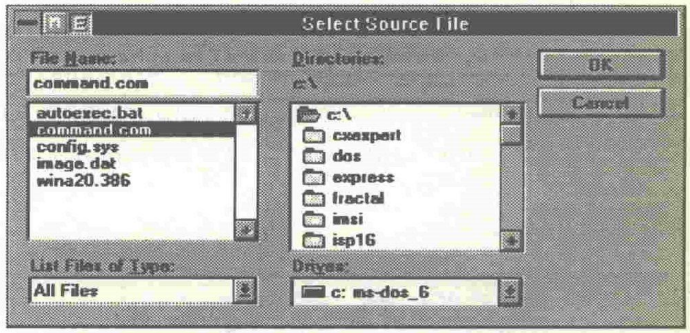
และเนื่องจากผู้เขียนต้องการให้ผู้อ่านได้เห็นถึงความรวดเร็วในการทำงานของ CIPHER.DLL ดังนั้นผู้เขียนจึงได้นำฟังก์ชันของไฟล์



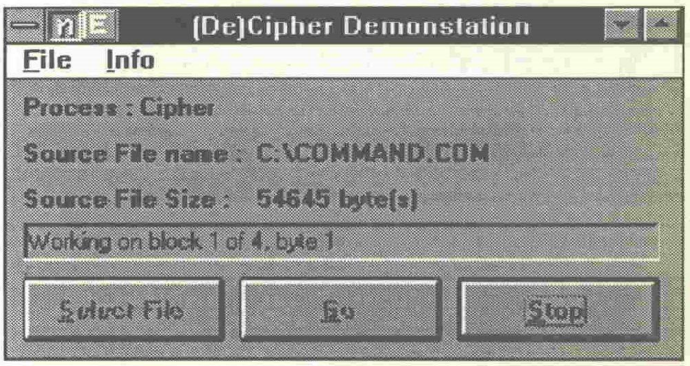
รูปที่ 2 แสดงการกำหนดรหัสผ่าน "Micro SE-ED"



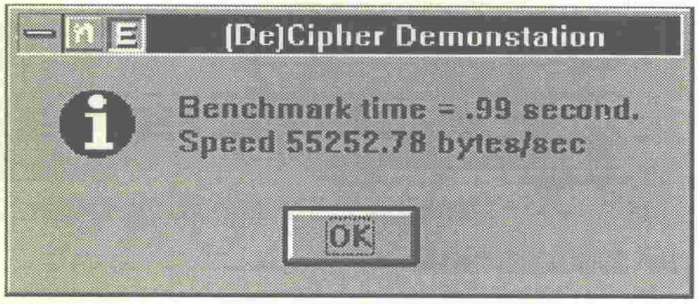
รูปที่ 3 แสดงการกำหนดขนาดของบล็อกข้อมูล



รูปที่ 4 แสดงการเลือกไฟล์ COMMAND.COM)



รูปที่ 5 แสดงรายละเอียดของไฟล์ COMMAND.COM



รูปที่ 6 แสดงเวลาในการปฏิบัติการเข้ารหัสหรือถอดรหัสข้อมูล

ไลบรารี BENMARK.DLL มาใช้ในการจับเวลาการทำงานในขบวนการเข้ารหัสและถอดรหัสข้อมูล ดังในรูปที่ 6 ข้างต้น เป็นการรายงานเวลาที่ใช้ในการเข้ารหัสไฟล์ COMMAND.COM ที่มีขนาด 54,654 ไบต์ ปรากฏว่าใช้เวลาในการเข้ารหัสข้อมูลเท่ากับ 0.99 วินาที หรือความเร็วในการทำงานเท่ากับ 55,252.78 ไบต์ต่อวินาที ซึ่งผู้อ่านจะเห็นว่าโปรแกรมสามารถทำงานได้รวดเร็วมาก และเท่าที่ผู้เขียนทำการทดสอบพบว่ามันสามารถทำงานได้เร็วกว่าโค้ดที่เขียนด้วย VB/Win ประมาณ 2 - 3 เท่าเลยทีเดียว

สำหรับผู้อ่านที่ต้องการโปรแกรม DCIPHER.EXE ไฟล์ไลบรารี CIPHER.DLL และ BENMARK.DLL ก็สามารถสั่งซื้อได้ทางซอฟต์เมล์ประจำฉบับนี้ ส่วนซอร์สโค้ดของโปรแกรม DCIPHER ผู้เขียนก็ขอยกยอดไปยังฉบับหน้าก็แล้วกันน่ะครับ พบกันใหม่ฉบับหน้าสวัสดีครับ

เทคนิคประจำคอลัมน์

ถ้าหากผู้อ่านทำการถอดรหัสข้อมูลของไฟล์ แล้วข้อมูลที่ได้ไม่ตรงกับต้นฉบับของเดิม อันเนื่องมาจากการกำหนดรหัสผ่านไม่ตรงกัน หรือกำหนดขนาดของบล็อกข้อมูลไม่เท่ากัน ผู้อ่านสามารถทำการกู้ข้อมูลก่อนที่จะทำการถอดรหัสข้อมูลกลับคืนมาได้ โดยการทำการเข้ารหัสข้อมูลไฟล์ดังกล่าวด้วยรหัสผ่านและขนาดของบล็อกข้อมูลที่ทำการถอดรหัสข้อมูลล่าสุดเท่านั้น

จากนั้นแล้วจึงทำการถอดรหัสข้อมูลของไฟล์ใหม่อีกครั้งด้วยรหัสผ่านและขนาดของบล็อกข้อมูลที่ถูกต้อง

โบราณว่า ช้า ๆ ได้พร้าเล่มงาม แต่ถ้าเป็นยุคนี้แล้วคงจะกลายเป็นช้า ๆ เสียพร้าเล่มงาม ยิ่งถ้าหากเป็นเทคโนโลยีในยุคไอทีแล้ว ช้าไม่ได้ค่ะ วันนี้มีรายละเอียดเกี่ยวกับความล่าช้าในการจำหน่ายซอฟต์แวร์ หากนับแต่วันประกาศตัวสินค้าก็เท่ากับผู้ขายรับปากเป็นมั่นเป็นเหมาะในการจัดจำหน่ายสินค้า แต่เมื่อถึงกำหนดวางตลาดก็มีอันต้องเลื่อนออกไป เราลองมาดูว่าระยะเวลาที่เลื่อนออกไปของแต่ละบริษัทเป็นอย่างไรบ้าง จะช้ามากน้อยอย่างไรดูรายละเอียดข้างล่างนี้เลยค่ะ

ผลิตภัณฑ์	กำหนดการณ์	วางตลาด	จำนวนเดือน (ที่คลาดเคลื่อน)
dBase for Windows	5/91	8/94	40
Lotus 1-2-3/G	4/87	3/90	36
ScriptX	6/92	12/94	31
Windows NT	4/91	7/93	28
Lotus 1-2-3/Mac	10/87	2/91	27
System 7.0	5/89	5/91	25
Windows 1.0	11/83	11/85	25
Lotus Notes	2/88	12/89	23
NextStep 3.0 DB Kit	12/90	9/92	22
Windows 95	12/93	8/95	21
Telescript	2/93	9/94	20
Windows 3.0	2/89	5/90	16
Ashton Tate SQL Server	1/88	4/89	16
Newton	5/92	7/93	15
NetWare 4.0	2/92	4/93	15
Novell DOS 7.0	3/93	3/94	13
Word 6.0 for Macintosh	10/93	9/94	12
cc:Mail wireless client	10/92	9/93	12
Quattro Pro for Windows	10/91	9/92	12
Envoy PDA	3/94	1/95	11
Power Macintosh	5/93	3/94	11
InForms	10/92	8/93	11
Microsoft Network	11/94	8/95	10
3DO	1/93	10/93	10
Zoomer	1/93	0/93	10
WordPerfect Office 4.0	10/92	6/93	9
Borland Delphi	8/94	3/95	8
SCO Win-tif	10/93	5/94	8
Adobe Acrobat	11/92	6/93	8
UnixWare	6/92	1/93	8
OpenDoc	6/93	ปลายปี 95	27
Copland	3/94	1996	ไม่ทราบจำนวน
Gershwin	3/94	1996	,,
Microsoft Exchange	6/94	1995	,,